REMARKS

I. STATUS OF THE CLAIMS

It is respectfully submitted that claims 1-18 are pending and under consideration.

II. REJECTIONS OF CLAIMS 17 AND 18 UNDER 35 U.S.C. §103(a) AS BEING UNPATENTABLE OVER DEVINS ET AL. (U.S. PATENT NO. 6,762,761)

Claim 17 recites automatically recognizing an operation status of the computer system in which the operation status represents what process is currently under execution. The Office Action states on page 3 that "an operation status representing what process is currently under execution" is taught by column 5, lines 15-20 of Devins. The Applicant respectfully disagrees.

Column 5, lines 15-20, of Devins discusses that:

> Accelerator 30 includes status registers (represented generically by status register 100) which contain status information relating to graphics operations performed by the accelerator under control of instructions in a captured program. The status registers also include status information relating to events on a hardware device external to accelerator 30.

Devins also discusses that it is "[a] captured program [that] may include instructions for causing the graphics processor to monitor the status information in a status register." Column 2, lines 47-49, of Devins. Thus, in other words, the graphics processor in Devins may monitor status information at the behest of a captured program, the **status information relating to graphics operations**. Such graphics operations typically comprise "basic rendering functions for rendering primitives, such as lines, circles and polygons." See column 1, lines 38-42, of Devins. On the other hand, claim 17 recites that the operation status represents what **process is currently under execution**.

Page 3 of the Office Action cites column 3, line 58, to column 4, line 7, of Devins as disclosing automatically starting a job, determined based on the recognized predetermined operation status. However, the cited portion of Devins discloses a device driver 15 including a capture routine 50 and I/O directives causing the DLP 25 to monitor the status of register 100 in the accelerator 30. The device driver 15 merely delays execution of predetermined instructions until the status register contains specified status information.

In fact, according to the Summary of the Invention of Devins, the system is directed to allowing a programmer to code graphics applications to execute graphics operations without initiating host processor hardware interrupt handling routines, by monitoring the status indicator

in the graphics accelerator and issuing hardware instructions in the captured programs based on the status information in the indicator. See column 2, lines 55-61, of Devins. That is, after a program is captured, instructions therein can cause the program to wait for a hardware event in the status register before executing program instructions. See, for example, Fig. 4, item 300, of Devins.

Column 6, lines 42-53, of Devins shows a captured program executing by continuously polling for a triggering event in a hardware status register. When the polling shows that the triggering event has occurred, then the programmed operations are executed. See also column 8, lines 5-11, of Devins. In other words, the programmed operations are predetermined (since the program is already chosen) and merely wait for a triggering event before executing; but there is no discussion of automatically starting a job, *determined based on the recognized predetermined operation status*, as recited in independent claim 1, for example. That is, Devins does not teach or suggest determining which job to automatically start, based on a recognized status.

As stated on page 3 of the Office Action, "Devins does not explicitly teach controlling the operating of an operating system." The Office Action then contends that "Devins teaches (lines 59-63 column 8) the system of the invention is implemented within an operating system. Therefore one of ordinary skill in the art would conclude that the method of Devins is used to control the operation of an operating system." The Applicant respectfully disagrees.

Devins discusses that "[f]or implementation, the device driver code may reside on any computer-usable medium at a suitable location, for example, in a computer memory as a fully-linked subroutine of a host operating system." See column 8, lines 59-63, of Devins. In light of this discussion, the Applicant finds no basis for the Office Action's assertion that the method and system for graphics rendering discussed in Devins controls an operating system in any fashion. The cited section of Devins merely discusses that device driver code may be implemented as a subroutine – there is no indication that such device driver code controls the operating system. Thus, Devins fails to render claim 17 unpatentable under 35 U.S.C. § 103(a).

The above comments are specifically directed to claim 17. However, it is respectfully submitted that the comments would be helpful in understanding various differences of various other claims over the cited art.

In view of the above, it is respectfully submitted that the rejection is overcome.

III. REJECTION OF CLAIMS 1-16 UNDER 35 U.S.C. § 103(a) AS BEING UNPATENTABLE OVER DEVINS ET AL. (U.S. PATENT NO. 6.762.761) IN VIEW OF MATSUMOTO (U.S. PATENT NO. 5,835,765)

Claim 1 recites preparing or deleting various kinds of files that show various operation statuses of the computer system in which an operation status represents what process is currently under execution. Claim 16 recites somewhat similar features. As the Office Action states on page 5, "Devins further does not explicitly teach storing operation statuses as files." The Office Action states that Matsumoto discloses this feature, citing column 2, lines 51-67. The Applicant respectfully disagrees.

The Office Action states on page 5 that "Matsumoto teaches (lines 51-67 column 2) a system of computer operation management system wherein operation statuses of running application programs are stored as log files." The cited section of Matsumoto discusses that:

> This computer operation management system comprises a program definition file predefined by the user to record the maximum number of simultaneously executable programs for each program category, the execution priority of the program, the error recovery procedure, programs that execute by communication with other programs on the network, and information about the post-process to be executed after the program terminates; a queried program record file for recording the information of queued programs waiting to be executed; and a log file for recording log data when a program starts or ends, and when an error occurs.

On the other hand, claim 1 recites that **preparing or deleting various kinds of files** that show various operation statuses of the computer system in which an operation status represents **what process is currently under execution**. Matsumoto fails to teach preparing or deleting various kinds of files. It appears from the discussion in Matsumoto that only a single log file is maintained. Further, the log file in Matsumoto simply records log data in a log file when a program starts or ends, and when an error occurs – it does not record an operating status representing which **process** is currently under execution. The program definition file and queried program record file also fail to represent what process is currently under execution.

As stated on page 3 of the Office Action, "Devins does not explicitly teach controlling the operating of an operating system." The Office Action then contends that "Devins teaches (lines 59-63 column 8) the system of the invention is implemented within an operating system. Therefore one of ordinary skill in the art would conclude that the method of Devins is used to control the operation of an operating system." The Applicant respectfully disagrees.

Devins discusses that "[f]or implementation, the device driver code may reside on any computer-usable medium at a suitable location, for example, in a computer memory as a fully-

linked subroutine of a host operating system." See column 8, lines 59-63, of Devins. In light of this discussion, the Applicant finds no basis for the Office Action's assertion that the method and system for graphics rendering discussed in Devins controls an operating system in any fashion. The cited section of Devins merely discusses that device driver code may be implemented as a subroutine – there is no indication that such device driver code **controls** the operating system.

Furthermore, Devins does not disclose the use of various operation statuses as defined by the present invention recited in independent claim 1. That is, Devins does not teach or suggest "recognizing a predetermined operation status of the computer system, depending on whether a file corresponding to the predetermined operation status exists within the memory section or not."

According to the Abstract in Devins, status information relates to a plurality of graphics operations performed by a graphics accelerator, and does not relate to operation statuses of a computer system, as recited in independent claim 1, for example. The Examiner notes on page 5 of the Office Action that Devins does not explicitly teach controlling the operating of the operating system (within a computer system), but states that the system of Devins is implemented within an operating system. However, it is respectfully submitted that the Examiner has not cited prior art that teaches recognizing the operation status "of the computer system", as recited in independent claim 1. In contrast, the Examiner has merely cited a portion of Devins that discusses operations performed by a graphics accelerator, which is not synonymous to the computer system of independent claim 1.

Page 5 of the Office Action cites column 3, line 58, to column 4, line 7, of Devins as disclosing automatically starting a job, determined based on the recognized predetermined operation status. However, the cited portion of Devins discloses a device driver 15 including a capture routine 50 and I/O directives causing the DLP 25 to monitor the status of register 100 in the accelerator 30. The device driver 15 merely delays execution of predetermined instructions until the status register contains specified status information.

In fact, according to the Summary of the Invention of Devins, the system is directed to allowing a programmer to code graphics applications to execute graphics operations without initiating host processor hardware interrupt handling routines, by monitoring the status indicator in the graphics accelerator and issuing hardware instructions in the captured programs based on the status information in the indicator. See column 2, lines 55-61, of Devins. That is, after a program is captured, instructions therein can cause the program to wait for a hardware event in the status register before executing program instructions. See, for example, Fig. 4, item 300, of Devins.

Column 6, lines 42-53, of Devins shows a captured program executing by continuously polling for a triggering event in a hardware status register. When the polling shows that the triggering event has occurred, then the programmed operations are executed. (See also column 8, lines 5-11, of Devins). In other words, the programmed operations are predetermined (since the program is already chosen) and merely wait for a triggering event before executing; but there is no discussion of automatically starting a job, *determined based on the recognized predetermined operation status*, as recited in independent claim 1, for example. That is, Devins does not teach or suggest determining which job to automatically start, based on a recognized status.

An operation status represents what process is currently under execution. Devins merely discusses that it is "[a] captured program [that] may include instructions for causing the graphics processor to monitor the status information in a status register." Column 2, lines 47-49, of Devins. The graphics processor in Devins may monitor status information at the behest of a captured program, the **status information relating to graphics operations**. Such graphics operations typically comprise "basic rendering functions for rendering primitives, such as lines, circles and polygons." See column 1, lines 38-42, of Devins. On the other hand, claim 17 recites that the operation status represents what **process is currently under execution**. Thus, Devins and Mastumoto, both individually and in combination, fail to render claim 1 unpatentable under 35 U.S.C. § 103(a).

The above comments are specifically directed to claim 1. However, it is respectfully submitted that the comments would be helpful in understanding various differences of various other claims over the cited art.

***

Dependent claims 4 and 5 inherit the patentable features of independent claim 1 and, thus, patentably distinguish over the prior art for the reasons set forth above. However, dependent claims 4 and 5 further recite, "the starting of the predetermined job is determined based on whether a plurality of the files exist or not within the memory section." See, for example, page 4, lines 3-8, and page 8, lines 16-18, and Fig. 1, reference numeral S12, of the present application.

In rejecting these features, the Examiner cites column 3, line 58, to column 4, line 7, of Devins, which merely states that a device driver 15 including a capture routine 50 and I/O directives causes the DLP 25 to monitor the status of register 100 in the accelerator 30. The device driver 15 simply delays execution of predetermined instructions until the status register contains specified status information.

However, the files, according to embodiments of the present invention, are prepared or deleted according to various operation statuses of the computer system, and the prepared files are stored in the memory. According to dependent claims 4 and 5, when a plurality of such files exists, the predetermined job is started.

It appears that the Examiner equates storing the executable instructions of Devins in the memory ready to be executed, to the plurality of files, according to claims 4 and 5. However, the executable instructions of Devins (which the Examiner also equates to the automatically-started job of the present invention), should not be considered synonymous to a plurality of the files, according to embodiments of the present invention. As stated above, the files of dependent claims 4 and 5 refer to the files prepared or deleted according to various operation statuses of the computer system and stored in the memory of the computer system. In fact, the executable instructions of Devins are not prepared according to an operation status of any computer system, but are merely predetermined and awaiting execution when appropriate status information is stored in the status register.

The above comments are specifically directed to claims 4 and 5. However, it is respectfully submitted that the comments would be helpful in understanding various differences of various other claims over the cited art.

In view of the above, it is respectfully submitted that the rejection is overcome.

IV. CONCLUSION

In accordance with the foregoing, it is respectfully submitted that all outstanding objections and rejections have been overcome and/or rendered moot. Further, all pending claims patentably distinguish over the prior art. There being no further outstanding objections or rejections, it is submitted that the application is in condition for allowance. An early action to that effect is courteously solicited.

Finally, if there are any formal matters remaining after this response, the Examiner is requested to telephone the undersigned to attend to these matters.

If there are any additional fees associated with filing of this Amendment, please charge the same to our Deposit Account No. 19-3935.

Respectfully submitted,

STAAS & HALSEY LLP

Date: 9-27-2007

By: _____

Michael A. Leonard II
Registration No. 60,180

1201 New York Avenue, N.W., 7th Floor
Washington, D.C. 20005
Telephone: (202) 434-1500
Facsimile: (202) 434-1501